**Machine Learning Assignment 2019**

Group:

Simon Rosen- 1391995

Christopher Pillay-1362077

Robel Abraham-1348744

Kimay Ramnarain-1370763

**Question 1**

**Question 2**

**Logistic Regression**

To split the data into the training, validation and test sets we made use of the ‘’train\_test\_split’’ function and applied it twice to the dataset, we produced the training data and test data first then further split the original training data into another set of training data and validation data, in both cases our test size was 20% for the splits. We then proceeded to clean the input data of our headlines by making use of regular expressions to remove all punctuation and convert the headlines to lowercase. To normalize and create our bag of words we used the TF-IDF vectorizer with its analyzer set to ‘word’.

**Naive Bayes**

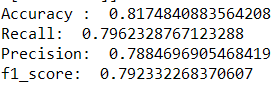
In order to split the data into training data, testing data and validation data, the 'test\_train\_split' function was used to split the data into training and testing data(80/20 split, respectively). In order to pre-process the data regular expressions were used to remove punctuation from the headlines so that we could analyse the words accurately.

**Question 3**

**Logistic Regression**

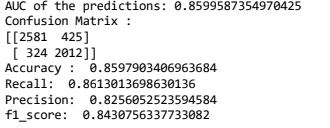
For Logistic Regression we created a class with all the required functions such as the add intercept function which set up our feature matrix by adding a column of 1’s to the front of the matrix, the sigmoid function which we made use of to generate our probabilities, the loss function which helps us find the best thetas for our model by measuring how well the algorithm performed with those thetas, a fit function to perform gradient descent and finally a predict and predict probability function which make use of the sigmoid function and an input threshold to

classify our results.

When we fitted the data to the model we used a learning rate of 0.5 since lower learning rates yielded lower rates in accuracy, our number of iterations was set at 10000 since low iterations yielded lower rates in accuracy and a higher number of iterations became computationally expensive. Our intercept was set to True since we wanted the constant present in our feature matrix. We set our threshold (determines whether a probability of a predicted value belongs to class 0 or 1) to a value of 0.5 as this gave us the highest accuracy when we tested values such as 0.3, 0.4, 0.6, 0.7 (See image ThresholdChoice)

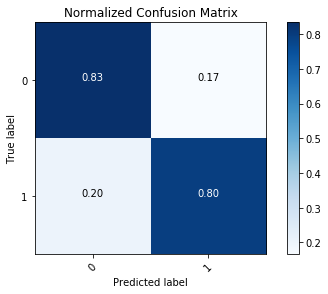
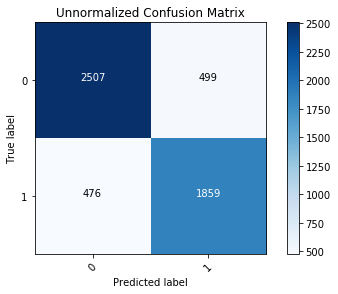
**Naive Bayes**

For Naive Bayes we created several functions to help make predictions. The first being the get\_text function, which gathered all headlines with the same classifications (0 for not sarcastic and 1 for sarcastic headlines). We then implemented the count\_text function, which returns the counter for the amount of times a word occurs. The get\_y\_count function was implemented in order to obtain the number of items in each class(0 or 1). These functions were implemented in order to assist with making predictions and calculating probabilities, which are needed when implementing Naive Bayes.

The main functions involved in classifying the headlines were the make\_decison and make\_class\_preditctions. The former of which was used to classify each word in each headline as either sarcastic or not sarcastic, we also applied smoothing in this function in order to multiplying/dividing by a zero when performing calculations. The latter of the two functions made the final decision on the whole headline based on which there were more sarcastic or non-sarcastic words in the headline. We set our alpha value for smoothing to 1 as this yielded the greatest accuracy, as seen in the results below:

**Question 4**

**Logistic Regression**

By looking at our unnormalized confusion matrix and accuracy we can see that we have a score of 0.817, this means that in our test data set, roughly 82% of the headlines were correctly predicted, which is quite a high score for our model. Our recall score was 0.796 which implies that approximately 80% our positive data was classified correctly, and our precision score was 0.788 implies that approximately 79% of predicted positives were actually positive. Therefore, a high recall and precision implies that the model will return a high percentage of results that are correctly classified. Our f1 score which is also known as our harmonic mean of the recall and precision score is 0.79, which implies that our model is a reasonable classifier. These scores reflect the best performance of our model for logistic regression which were achieved by testing and changing our hyperparameters. Although we recommend that stochastic gradient descent be used instead of our regular implementation of gradient descent as it will speed up computation and this would allow more iterations when the model is trained to improve the results of the classifier.

**Naive Bayes**

Our Naive Bayes algorithm yielded an accuracy of 0.859, the algorithm that we implemented predicted the headlines correctly 85% of the times. From our recall value of 0.86, we see that 86% of the headlines that were sarcastic were correctly classified as sarcastic. Our precision score was 0.82 which implies that 82% of all headlines that were predicted to be sarcastic, were in fact sarcastic. These high statistics tell us that a high percentage of the headlines were correctly classified. Our f1 score was 0.84 which implies that our Naive Bayes model is a good classifier. We recommend calculating probabilities that are needed for this function, such as standard deviation, mean, etc…for each class of words in separate functions as it allows calculations to be performed with a bit more ease and makes error tracing easier.